README

Introduction

This project is a sentiment analysis task that classifies movie reviews as either expressing positive or negative emotions. We built an ensemble model using both MLP (LSTM+CNN) and Naive-Bayes models. The input to the model is the original text comments in the form of a sequence of words. The output of the model is an integer between 0 to 9, representing the level of sentiment. We further classify the output, with scores 1-4 representing negative and 7-10 representing positive sentiment.

Model

- Figure

For the MLP model (Figure 1), we feed input data with shape (batch size, sentence length) into an embedding layer. The output of the embedding layer is passed through a dropout layer to mitigate overfitting. Then, we apply a one dimensional convolution layer with a Rectified Linear Unit (ReLU) activation function before performing max-pooling. Next, we use a Long Short-Term Memory (LSTM) layer followed by two linear fully-connected layers to produce the final output.

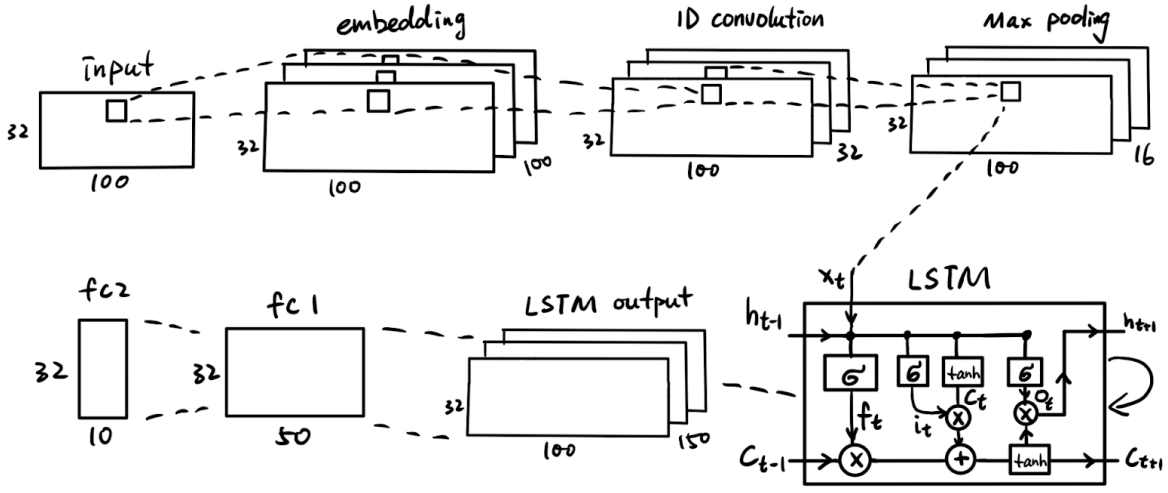


Figure 1. MLP model figure

The Naive Bayes model does not have any hyperparameters, therefore, there is no need for a figure to visualize it. This model analyzes all the data from movie reviews in the training data to predict whether a given review is positive or negative. We believe that certain words can strongly indicate a positive or negative sentiment in a review. Thus, when making the final prediction, our team assigns greater weight to the prediction result from the simple Naive Bayes model. As the way of processing input is different for Naive Bayes model and LTSM model, we use a separate vocabulary set for Naive Bayes model.

- Parameters

    - Naive Bayes

The Naive Bayes model has two parameter variables which are theta and pi. The shape of theta should be (vocabulary size, 10). While the vocabulary size is 42268, so the shape of theta is (42268, 10). The shape of pi should be 10. Therefore, the number of parameters in the Naive Bayes model is 42268 * 10 + 10 = 422690.

    - LSTM

The MLP model consists of an embedding layer, a 1D convolutional layer, an LSTM layer, and two fully-connected layers. The embedding layer has a weight matrix with trainable parameters, and no bias is set. The matrix's shape is the vocabulary size by embedding size, which we set as 41456 and 100 respectively. Hence, the number of parameters in the embedding layer is 41456 * 100 = 4145600.

In the convolutional layer, the number of parameters is determined by the size of input channels, output channels, and the kernel size, which we set as 100, 32, and 3 respectively. Therefore, the number of parameters of this layer is (in_channels * kernel_size * out_channels) + bias for each output channel, which is 100 * 3 * 32 + 32 = 9632.

In the LSTM layer, considering the inside computation of input gate, forget gate, output gate, and cell state, each gate has a pair of weight matrix and bias. For instance, in the input gate, the first weight matrix has a size of hidden size by input size, and the second weight matrix has a size of hidden size by hidden size. There is also a bias vector with hidden size of parameters. Summing the number of parameters in four gates, we have 4 * (150 * 50 + 150 * 150 + 150) = 120600.

Lastly, there are two linear fully-connected layers. The first fully-connected layer has an input size of hidden size, an output size of 50, and a bias for each output. Hence, the number of parameters is 150 * 50 + 50 = 7550. The second fully-connected layer has an input size of 50, an output size of target classes, and a bias for each target class. Therefore, the number of parameters is 50 * 10 + 10 = 510. Hence, the total number of parameters in two fully-connected layers is 7550 + 510 = 8060.

In conclusion, the total number of parameters in the entire model is 422690 + 4145600 + 9632 + 120600 + 8060 = 4706582.

- Examples from Test Set

One of the successful example is:
list(['movi', 'first', 'franc', 'hometown', 'year', 'today', 'still', 'rememb', 'black', 'white', 'frame', 'especi', 'black', 'tens', 'scari', 'sneak', 'attack', 'mountain', 'soldier', 'prison', 'forc', 'fight', 'captor', 'battl', 'uncov', 'wagon', 'gatl', 'fire', 'fight', 'chief', 'happi', 'breath', 'sinc', 'would', 'today', 'fabul', 'moment', ''])]
result: 8 target: 8

One of the unsuccessful example is:
list(['inglori', 'basterd', 'make', 'apolog', 'forgiv', 'hold', 'bar', 'assault', 'sens', 'tarantino', 'offend', 'step', 'stereotyp', 'cliché', 'make', 'purest', 'great', 'maker', 'whose', 'clearli', 'interfear', 'power', 'tarantino', 'master', 'effortlessli', 'crank', 'immens', 'tension', 'suddenli', 'mix', 'laugh', 'moment', 'look', 'disgust', 'roll', 'around', 'laugh', 'either', 'roller', 'coaster', 'miss', 'everyon', 'unsur', 'german', 'certainli', 'tarantino', 'style', 'littl', 'swallow', 'never', 'simpli', 'respect', 'perform', 'christoph', 'waltz', 'oscar', 'ignor', ''])]
result: 1 target: 10

Data

- Source

IMDB dataset (http://ai.stanford.edu/~amaas/data/sentiment/)

- Statistics Summary

We began our analysis by examining the emotion levels of the dataset. The resulting graph is shown below (Figure 2). Since both the mean and median values are close to 5, indicating a neutral emotional state, we can conclude that the data is unbiased in terms of sentiment, and can therefore be used for training our model.

Furthermore, we observed that there were a significant number of comments with extreme sentiment values (such as 1 and 10). By incorporating more extreme sentiment comments in the training data, our model will be exposed to a wider range of emotional expressions, which may help it to better distinguish between positive and negative sentiment in test data. This, in turn, should improve the overall accuracy of our model.

```
total number of scores: 25000
1 occurrence count:  5100
2 occurrence count:  2284
3 occurrence count:  2420
4 occurrence count:  2696
7 occurrence count:  2496
8 occurrence count:  3009
9 occurrence count:  2263
10 occurrence count:  4732
negative sum:  12500
positive sum:  12500
mean:  5.47772
median:  5.5
most occurrence number:  1
```

Figure 2. training data statistics

For the word analyze, we discussed it into two parts: keep stopwords and remove stopwords. Stopwords mean these words are commonly used in a language but not meaningful. With stopwords, the most common 10 words are ‘this’, ‘that’, ‘movie’, ‘with’, ‘film’, ‘have’, ‘they’, 'like’, ‘from’, ‘there’. We can see these most common 10 words did not contribute much emotional bias to the dataset, so we decided to remove stopwords. Then, the most common 10 words became 'movi', 'film', 'like', 'thi', 'time', 'good', 'watch', 'make', 'stori', 'charact', and the size of vocab is 41456.

- Transformation & Split

    - Part 1: Downloaded files to input txt file

The original dataset of 50,000 movie reviews was divided into two folders: train data and valid data, each with 25,000 reviews. These folders were further divided into positive and negative folders, each containing 12,500 text files. Each text file represented a complete movie review, with the file name indicating the score given to the movie on a scale of 1 to 10.

To make it easier to use as input for our model, we processed the raw dataset in a local folder. We wrote all the movie reviews in the train data folder into a new text file named train_data.txt, with the scores represented by each review in the first character of each sentence. This enabled easy differentiation between the review scores and the review text when reading the reviews in the model. Similarly, all the reviews in the valid data folder were written into a text file named valid_data.txt for input into the model.

To reduce the running cost of the model, we captured only the first 100 words of each review and entered them into the train_data and valid_data databases. In most cases, the first 100 words were sufficient to convey the essence of the review.

    - Part 2: read txt file into model input

We first remove line breaks (symbol "
") and punctuations when reading each line from the txt file. Additionally, to reduce the vocabulary size, we apply several criteria to process the data, such as removing stop words, digits, misspelled words, and stemming which means reducing words into the root form. Next, we set the maximum length of each sentence to 100 words, because we find out that emotions are clearly expressed within 100 words in the dataset. For sentences that are shorter than 100 words, we pad the remaining space with empty strings, and for sentences that exceed 100 words, we keep the first 100 words. We place the target labels at the beginning of each review, resulting in a dataset with a shape of N * 2, where each row has an integer label followed by a list of words.

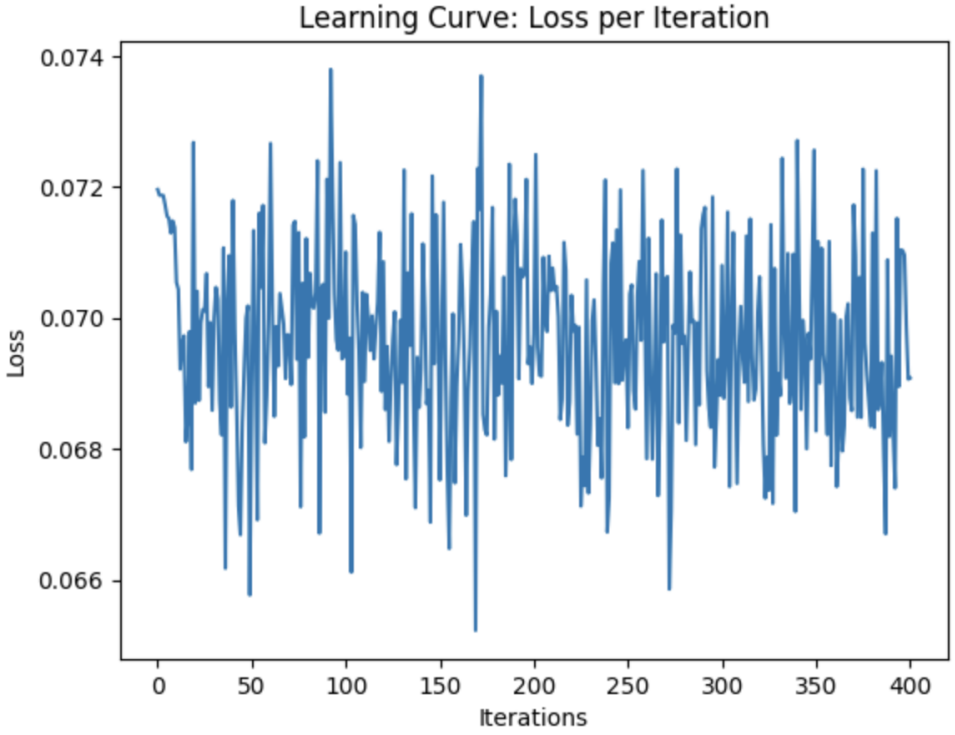
In the get_batch() function, we split the input vector x and target vector t by batch size, converting each word into an index in the vocabulary. Since we use the nn.Embedding function in our MLP model, the input sequence of indices does not need to be converted into one-hot vectors.

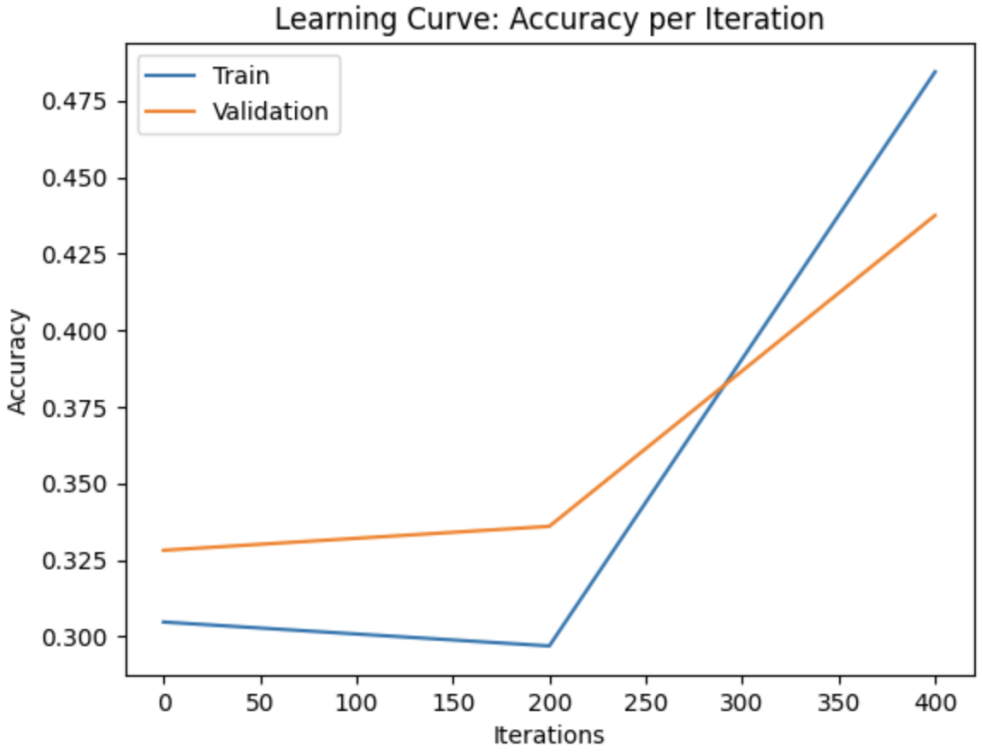
- Part 3: data split

To train our model, we used the provided training set and split the provided test dataset into validation and test sets. We split the dataset in a 50%-25%-25% ratio for training, validation, and test sets, respectively.

Training

- Training curve





Hyperparameter Tuning

In the hyperparameter tuning process, we focused on finding the best combination of hyperparameters that could improve the performance of our model.

We started by adding new layers for the MLP model. We explored different number of fully-connected layers with 1 and 2 layers and evaluated the model's performance on the validation set. We found that our model with 2 fully-connected layers performed the best, so we used 2 fully-connected layers in the final model.

Then we tested different learning rate values, including 0.001, 0.01, and 0.1. We found that the validation accuracy decreased and was unstable. We think the learning rates with 0.01 and 0.1 are too high to miss the minimum of the loss function, so we decided to use 0.001 as the learning rate of our model.

Next, we experimented with different batch sizes, including 16, 32, and 64, and found that a batch size of 32 produced the best results on the validation set.

We also tried different hyperparameters for the MLP model and Naive Bayes model, but did not observe any significant improvement in the accuracy.

Overall, we believe that the hyperparameters we chose made sense for our model and dataset, as they produced the best performance on the validation set.

Results

- Quantitative Measures

We will use accuracy to measure the result.

- Quantitative and Qualitative Results

Our final model has exhibited a high level of performance with a test accuracy of approximately 82% (± 0.05). This result is particularly impressive as the model has been tested on a previously unseen test set, which we had no prior knowledge of, and yet it has consistently made accurate predictions. The success of our model is a testament to the effectiveness of the techniques we employed during the training process, as well as the quality of our dataset.

- Justification of Results

Since our task is to distinguish whether a film review is positive or negative, we need to choose models suitable for this task. After discussions in our group, we decided to use the naive Bayes model and RNN model.

The RNN model is suitable for processing input data in sequence, and it can also consider the context to make more accurate predictions for positive and negative reviews. For the naive Bayes model, the model calculates the probability of each word appearing in positive and negative reviews to determine whether each review is positive or negative. Both models are well-suited for our task, and we plan to combine them using ensemble methods to prevent overfitting.

For the RNN model, we created a large vocabulary to store words in the training data and analyze the context to determine the positive or negative of the review. However, we set a maximum word limit of 100 words per comment, which may result in loss of information and some inaccurate predictions due to users' tendencies to promote before criticizing. Despite this, we believe that 100 words is the best balance between efficiency and accuracy. Therefore, we think the RNN model can handle this task perfectly.

For the naive Bayes model, we also created a large vocabulary vocab to store words in the training data and calculate the probabilities of each word's occurrence in positive and negative reviews. Since some words have strong associations with positive or negative reviews, such as "awesome" or "upset", we randomly selected 25,000 data points as the training set, and used the remaining 12,500 data points as the test set. We calculated a theta map with two different classes and the probabilities (pi) of each class. When we tested the accuracy of this model, we found it to be higher than 80%. Therefore, we decided to include this model in our final prediction results. Although naive Bayes models do not have hyperparameters, their accuracy is affected by the composition of the training set. If the training set contains more data points with detailed reviews, the model's accuracy will be higher. Therefore, the simple Bayesian model is more susceptible to overfitting than other models. To prevent overfitting, we adjusted the weight of the Bayesian model's prediction results in the final results, and used ensembles to help us overcome this issue.

Ethical Consideration

One ethical benefit is that it can help audiences know the rating of a movie without being spoiled. Audiences may leave pieces of plot in review on social media, so other audiences can use our model to evaluate the movie based on these reviews without knowing the plot of the movie by accident. However, the dataset may not be balanced in terms of demographics,

leading to a biased problem. As different movie themes may appeal to distinct groups of people, they may elicit various emotional responses in reviews that could influence the movie's ratings.

Movie producers can use our model to understand the public's evaluation of their movies based on scattered reviews on social media that may not include a formal rating. This can guide producers to make future decisions regarding marketing and production. However, the ethical issue is movie producers may increasingly rely on sentiment analysis for evaluating audience sentiment, but it should be used in conjunction with other factors and not as the sole determinant.

Also, there are limitations of our model. One limitation is lack of context awareness. Some sentences may have irony or sarcasm. It is hard to determine the real meanings of those sentences without considering the background of those sentences, hence it is hard to classify the movie reviews as positive or negative. Another limitation is lack of topic knowledge. Since our dataset doesn't include the topic of movies, our model may not accurately generate appropriate ratings for specific movie topics. For example, 'It makes me cry' in a horror movie may mean something positive.

The limitation of our dataset is lack of other forms of comments. For example, many comments will have emojis, pictures, but our dataset only concludes text. Those emojis and pictures also express a lot of meanings, and even some text content with emojis may indicate jokes, irony, etc. Therefore, the lack of these forms of comments will make our rating not entirely accurate.

Authors

Jialin CAI: process model input, write the LSTM + CNN model, analyze output of models, (in README:) write introduction part, model figure part, model parameters part, write data source part, write data split part.
Ziyang Qu: write data analysis code, tuning hyperparameters, training curve,(in README:) write data summary part, tuning hyperparameters, training curve, write ethical consideration part, write authors part, model examples part, establish github, submit project.
Yifan QIN: find dataset source, write get accuracy code, plot training curve, (in README:) Justification of Results and ensembles two models.
Zeyang WANG: write Naive-bayes model, write data transform code, write quantitative measures part, write quantitative and qualitative results part, write justification of results part.